# Static files (CSS, JavaScript, Images)

# https://docs.djangoproject.com/en/4.0/howto/static-files/

STATIC\_URL = 'static/'

MEDIA\_URL='/images/'

MEDIA\_ROOT=BASE\_DIR/'static'

STATICFILES\_DIRS=[

BASE\_DIR/'static'

]

Use load static before

{% load static %}

<!doctype html>

<html lang="en">

<head>

<meta charset="utf-8">

<meta name="viewport" content="width=device-width, initial-scale=1">

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC" crossorigin="anonymous">

<title>

{% block title %}

{% endblock title %}

</title>

<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">

<link rel="stylesheet" href="{% static 'css/style.css' %}">

</head>

<body>

{% include 'ekart/supportfiles/navbar.html' %}

{% block content %}

{% endblock content %}

{% block scripts %}

{% endblock scripts %}

{% include 'ekart/supportfiles/footer.html' %}

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.js" integrity="sha384-MrcW6ZMFYlzcLA8Nl+NtUVF0sA7MsXsP1UyJoMp4YLEuNSfAP+JcXn/tWtIaxVXM" crossorigin="anonymous"></script>

</body>

</html>

Use static before for loading images too  
  
{% load static %}

<div id="carouselExampleIndicators" class="carousel slide" data-bs-ride="carousel">

<div class="carousel-indicators">

<button type="button" data-bs-target="#carouselExampleIndicators" data-bs-slide-to="0" class="active" aria-current="true" aria-label="Slide 1"></button>

<button type="button" data-bs-target="#carouselExampleIndicators" data-bs-slide-to="1" aria-label="Slide 2"></button>

<button type="button" data-bs-target="#carouselExampleIndicators" data-bs-slide-to="2" aria-label="Slide 3"></button>

<button type="button" data-bs-target="#carouselExampleIndicators" data-bs-slide-to="3" aria-label="Slide 4"></button>

</div>

<div class="carousel-inner">

<div class="carousel-item active">

<img src="{% static 'images/1.jpg' %}" class="d-block w-100" alt="Slide 1">

</div>

<div class="carousel-item">

<img src="{% static 'images/2.jpg' %}" class="d-block w-100" alt="Slide 2">

</div>

<div class="carousel-item">

<img src="{% static 'images/3.jpg' %}" class="d-block w-100" alt="Slide 3">

</div>

<div class="carousel-item">

<img src="{% static 'images/4.jpg' %}" class="d-block w-100" alt="Slide 4">

</div>

</div>

<button class="carousel-control-prev" type="button" data-bs-target="#carouselExampleIndicators" data-bs-slide="prev">

<span class="carousel-control-prev-icon" aria-hidden="true"></span>

<span class="visually-hidden">Previous</span>

</button>

<button class="carousel-control-next" type="button" data-bs-target="#carouselExampleIndicators" data-bs-slide="next">

<span class="carousel-control-next-icon" aria-hidden="true"></span>

<span class="visually-hidden">Next</span>

</button>

</div>

Under main combining separate html content code with MainLayout.html by using “Include” (navbar, footer)

{% load static %}

<!doctype html>

<html lang="en">

<head>

<meta charset="utf-8">

<meta name="viewport" content="width=device-width, initial-scale=1">

<link href="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/css/bootstrap.min.css" rel="stylesheet" integrity="sha384-EVSTQN3/azprG1Anm3QDgpJLIm9Nao0Yz1ztcQTwFspd3yD65VohhpuuCOmLASjC" crossorigin="anonymous">

<title>

{% block title %}

{% endblock title %}

</title>

<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/4.7.0/css/font-awesome.min.css">

<link rel="stylesheet" href="{% static 'css/style.css' %}">

</head>

<body>

{% include 'ekart/supportfiles/navbar.html' %}

{% block content %}

{% endblock content %}

{% block scripts %}

{% endblock scripts %}

{% include 'ekart/supportfiles/footer.html' %}

<script src="https://cdn.jsdelivr.net/npm/bootstrap@5.0.2/dist/js/bootstrap.bundle.min.js" integrity="sha384-MrcW6ZMFYlzcLA8Nl+NtUVF0sA7MsXsP1UyJoMp4YLEuNSfAP+JcXn/tWtIaxVXM" crossorigin="anonymous"></script>

</body>

</html>

For implementing registration and login through mysql  
  
DATABASES = {

'default': {

'ENGINE': 'django.db.backends.mysql',

'NAME': 'testproject04',

'HOST':'192.168.1.241',

'USER':'test01',

'PASSWORD':'test@123',

'PORT':'3306'

}

}

For custom forms  
  
  
implement this under models.py  
  
from itertools import product

from django.db import models

from django.contrib.auth.models import User

import datetime

import os

then use it with your own forms.py or customform py file

from pyexpat import model

from django.contrib.auth.forms import UserCreationForm

from .models import User

from django import forms

class CustomUserForm(UserCreationForm):

username=forms.CharField(widget=forms.TextInput(attrs={'class':'form-control','placeholder':'Enter User Name'}))

email=forms.CharField(widget=forms.TextInput(attrs={'class':'form-control','placeholder':'Enter Email Address'}))

password1=forms.CharField(widget=forms.PasswordInput(attrs={'class':'form-control','placeholder':'Enter Your Password'}))

password2=forms.CharField(widget=forms.PasswordInput(attrs={'class':'form-control','placeholder':'Enter Confirm Password'}))

class Meta:

model=User

fields=['username','email','password1','password2']  
  
  
  
then implement the imports under views.py under create App  
  
from django.http import JsonResponse

from django.shortcuts import redirect, render

from ekart.forms import CustomUserForm

from . models import \*

from django.contrib import messages

from django.contrib.auth import authenticate,login,logout

import json  
  
  
  
  
then use it under the view defined as Def method like below (which contains both GET and POST logic)  
  
def register(request):

form=CustomUserForm()

if request.method=='POST':

form=CustomUserForm(request.POST)

if form.is\_valid():

form.save()

messages.success(request,"Registration Success You can Login Now..!")

return redirect('/login')

return render(request,"ekart/register.html",{'form':form})

then you must have register.html with the below logical code to implement forms elements like username field and password field so on to display to the user  
  
{% extends 'ekart/layouts/main.html' %}

{% block title %}

Registration | ShopKart

{% endblock title %}

{% block content %}

<div class="container" style="margin-top:70px;min-height:600px;">

<div class="row">

<div class="col-12">

<h4 class="mb-3"> New User Registration</h4>

<hr style="border-color:#b8bfc2;">

{% include 'ekart/supportfiles/message.html' %}

</div>

</div>

<section class="py-4" >

<div class="container">

<div class="row">

<div class="col-6">

<form method="post" action="">

{% csrf\_token %}

<div class="mb-4">

<label for="" class="form-label">User Name</label>

{{form.username}}

</div>

<div class="mb-4">

<label for="" class="form-label">User Email</label>

{{form.email}}

</div>

<div class="mb-4">

<label for="" class="form-label">Password</label>

{{form.password1}}

</div>

<div class="mb-4">

<label for="" class="form-label">Confirm Password</label>

{{form.password2}}

</div>

<button type="submit" class="btn btn-primary">Register</button>

</form>

<p class="my-2">Already user ? <a href="{% url 'login' %}">Login Now</a> </p>

</div>

<div class="col-6">

{% if form.errors.username %}

<label class="text-danger d-block">{{ form.errors.username }}</label>

{% endif %}

{% if form.errors.email %}

<label class="text-danger d-block">{{ form.errors.email }}</label>

{% endif %}

{% if form.errors.password1 %}

<label class="text-danger d-block">{{ form.errors.password1 }}</label>

{% endif %}

{% if form.errors.password2 %}

<label class="text-danger d-block">{{ form.errors.password2 }}</label>

{% endif %}

</div>

</div>

</div>

</section>

</div>

{% endblock content %}  
  
  
  
for creating and registering under given table logic you must for the is before running   
  
python Manage.py makemigrations  
python manage.py migrate  
  
above two will help you to create default tables under Admin logic which will be used for authentication  
  
if these methods followed then create super user with the help of  
  
python Manage.py createsuperuser  
then follow with instruction to name it with your own username, email if needed, then password.  
  
once superuser created login with the help of your\_ip:youport given by the following command  
  
python manage.py runserver

localhost:8000/admin  
use the superuser username and password created above.  
  
if user registers through registration page then it will be added to database and it can be viewed admin user which can be done through above steps